Integrating Semantic Knowledge with Web Usage Mining for Personalization

Honghua Dai and Bamshad Mobasher

School of Computer Science, Telecommunication, and Information Systems
DePaul University
243 S. Wabash Ave.
Chicago, Illinois, 60604

Phone: 312-362-5174
Fax: 312-362-6116
Email: mobasher@cs.depaul.edu

Keywords: Web usage mining, personalization, data mining, domain knowledge, ontologies, semantic Web mining

Integrating Semantic Knowledge with Web Usage Mining for Personalization

Abstract

Web usage mining has been used effectively as an approach to automatic personalization and as a way to overcome deficiencies of traditional approaches such as collaborative filtering. Despite their success, such systems, as in more traditional ones, do not take into account the semantic knowledge about the underlying domain. Without such semantic knowledge, personalization systems cannot recommend different types of complex objects based in their underlying properties and attributes. Nor can these systems possess the ability to automatically explain or reason about the user models or user recommendations. The integration of semantic knowledge is, in fact, the primary challenge for the next generation of personalization systems. In this chapter we provide an overview of approaches for incorporating semantic knowledge into Web usage mining and personalization processes. In particular we discuss the issues and requirements for successful integration of semantic knowledge from different sources, such as the content and the structure of Web sites for personalization. Finally, we present a general framework for fully integrating domain ontologies with Web Usage Mining and Personalization processes at different stages, including the preprocessing and pattern discovery phases, as well as in the final stage where the discovered patterns are used for personalization.

INTRODUCTION

With the continued growth and proliferation of e-commerce, Web services, and Web-based information systems, personalization has emerged as a critical application which is essential to the success of a Web site. It is now common for Web users to encounter sites that provide dynamic recommendations for products and services, targeted banner advertising, and individualized link selections. Indeed, nowhere is this phenomenon more apparent as in the business-to-consumer e-commerce arena. The reason is that, in today's highly competitive e-commerce environment, the success of a site often depends on the site's ability to retain visitors and turn casual browsers into potential customers. Automatic personalization and recommender system technologies have become critical tools, precisely because they help engage visitors at a deeper and more intimate level by tailoring the site's interaction with a visitor to her needs and interests.

Web personalization can be defined as any action that tailors the Web experience to a particular user, or set of users (Mobasher, Cooley, & Srivastava, 2000a). The experience can be something as casual as browsing a Web site or as (economically) significant as trading stocks or purchasing a car. Principal elements of Web personalization include modeling of Web objects (pages, etc.) and subjects (users), categorization of objects and subjects, matching between and across objects and/or subjects, and determination of the set of actions to be recommended for personalization. The actions can range from simply making the presentation more pleasing to anticipating the needs of a user and providing customized information.

Traditional approaches to personalization have included both content-based and user-based techniques. Content-based techniques use personal profiles of users and recommend other items or pages based on their content similarity to the items or pages that are in the user's profile. The underlying mechanism in these systems is usually the comparison of sets of keywords representing pages or item descriptions. Examples of such systems include Letizia (Lieberman, 1995) and WebWatcher (Joachims, Freitag, & Mitchell, 1997). While these systems perform well from the perspective of the end user who is searching the Web for information, they are less useful in e-commerce applications, partly due to the lack of server-side control by site owners, and partly because techniques based on content similarity alone may miss other types of semantic relationships among objects (for example, the associations among products or services that are semantically different, but are often used together).

User-based techniques for personalization, on the other hand, primarily focus on the similarities among users rather than item-based similarities. The most widely used technology user-based personalization is collaborative filtering (CF) (Herlocker, Konstan, Borchers, & Riedl, 1999). Given a target user's record of activity or preferences, CF-based techniques compare that record with the historical records of other users in order to find the users with similar interests. This is the so called *neighborhood* of the current user. The mapping of a visitor record to its neighborhood could be based on similarity in ratings of items, access to similar content or pages, or purchase of similar items. The identified neighborhood is then used to recommend items not already accessed or purchased by the active user. The advantage of this approach over purely content-based approaches which rely on content similarity in item-to-item comparisons is that it can capture "pragmatic" relationships among items based on their intended use or based on similar tastes of the users.

The CF-based techniques, however, suffer from some well-known limitations (Sarwar, Karypis, Konstan, & Riedl, 2000). For the most part these limitations are related to the scalability and efficiency of the underlying algorithms which requires real-time computation in both the neighborhood formation and the recommendation phases. The effectiveness and scalability of collaborative filtering can be dramatically enhanced by the application of Web usage mining techniques.

In general, *Web mining* can be characterized as the application of data mining to the content, structure, and usage of Web resources (Cooley, Mobasher, & Srivastava, 1997; Srivastava, Cooley, Deshpande, & Tan, 2000). The goal of Web mining is to automatically discover local as well as global models and patterns within and between Web pages or other Web resources. The goal of *Web usage mining*, in particular, is to capture and model Web user behavioral patterns. The discovery of such patterns from the enormous amount of data generated by Web and application servers has found a number of important applications. Among these applications are systems to evaluate the effectiveness of a site in meeting user expectations (Spiliopoulou, 2000), techniques for dynamic load balancing and optimization of Web servers for better and more efficient user access (Pitkow & Pirolli, 1999; Palpanas & Mendelzon, 1999), and applications for dynamically restructuring or customizing a site based on users' predicted needs and interests (Perkowitz & Etzioni, 1998).

More recently, Web usage mining techniques have been proposed as another user-based approach to personalization which alleviate some of the problems associated with collaborative filtering (Mobasher et al,, 2000a). In particular, Web usage mining has been used to improve the scalability of personalization systems based on traditional CF-based techiques (Mobasher, Dai, Luo, & Nakagawa, 2001; Mobasher, Dai, Luo, & Nakagawa, 2002).

However, the pure usage-based approach to personalization has an important drawback: the recommendation process relies on the existing user transaction data, thus items or pages added to a site recently cannot be recommended. This is generally referred to as the "new item problem". A common approach to revolving this problem in collaborative filtering has been to integrate content characteristics of pages with the user ratings or judgments (Claypool et al. 1999; Pazzani, 1999). Generally, in these approaches, keywords are extracted from the content on the Web site and are used to either index pages by content or classify pages into various content categories. In the context of personalization, this approach would allow the system to recommend pages to a user, not only based on similar users, but also (or alternatively) based on the content similarity of these pages to the pages user has already visited.

Keyword-based approaches, however, are incapable of capturing more complex relationships among objects at a deeper semantic level based on the inherent properties associated with these objects. For example, potentially valuable relational structures among objects such as relationships between movies, directors, and actors, or between students, courses, and instructors, may be missed if one can only rely on the description of these entities using sets of keywords. To be able to recommend different types of complex objects using their underlying properties and attributes, the system must be able to rely on the characterization of user segments and objects, not just based on keywords, but at a deeper semantic level using the domain ontologies for the objects. For instance, in a traditional personalization system on a university Web site might recommend courses in Java to a student, simply because that student has previously taken or shown interest in Java courses. On the other hand, a system that has knowledge of the underlying domain ontology, might recognize that the student should first satisfy the prerequisite requirements for a recommended course, or be able to recommend the best instructors for Java course, and so on.

An *ontology* provides a set of well-founded constructs that define significant concepts and their semantic relationships. An example of an ontology is a relational schema for a database involving multiple tables and foreign keys semantically connecting these relations. Such constructs can be leveraged to build meaningful higher level knowledge in a particular domain. Domain ontologies for a Web site usually include concepts, subsumption relations between concepts (concept hierarchies), and other relations among concepts that exist in the domain that the Web site represents. For example, the domain ontologies of a movie Web site usually includes concepts such as "movie," "actor," "director," "theater" etc. The genre hierarchy can be used to represent different categories of movie concepts. Typical relations in this domain may include, "Starring" (between actors and movies), "Directing", "Playing" (between theaters and movies), etc.

The ontology of a Web site can be constructed by extracting relevant concepts and relations from the content and structure of the site, through machine learning and Web mining techniques. But, in addition to concepts and relations that can be acquired from Web content and structure information, we are also interested in usage-related concepts and relations in a Web site. For instance, in an E-commerce Web site, we may be interested in the relations between users and objects which define different types of online activity, such as browsing, searching, registering, buying, and bidding. The integration of such usage-based relations with ontological information representing the underlying concepts and attributes embedded in a site allows for more effective knowledge discovery, as well as better characterization and interpretation of the discovered patterns.

In the context of Web personalization and recommender systems, the use of semantic knowledge can lead to deeper interaction of the visitors or customers with the site. Integration of domain knowledge allows such systems to infer additional useful recommendations for users based on more fine grained characteristics of the objects being recommended, and provides the capability to explain and reason about user actions.

In this chapter we present an overview of the issues related to and requirements for successfully integrating semantic knowledge in the Web usage mining and personalization processes. We begin by providing some general background on the use of semantic knowledge and ontologies in Web mining, as well as an overview of personalization based on Web usage mining. We then discuss how the content and the structure of the site can be leveraged to transform raw usage data into semantically-enhanced transactions that can be used for semantic Web usage mining and personalization. Finally we present a framework for more systematically integrating full-fledged domain ontologies in the personalization process.

BACKGROUND

Semantic Web Mining

Web mining is the process of discovering and extracting useful knowledge from the content, usage, and structure of one or more Web sites. Semantic Web mining (Berendt, Hotho, & Stumme, 2002) involves the integration of domain knowledge into the Web mining process.

For the most part the research in Semantic Web mining has been focused in application areas such as Web content and structure mining. In this section, we provide a brief overview and some examples of related work in this area. Few studies have focused on the use of domain knowledge in Web usage mining. Our goal in this chapter is to provide a road map for the integration of semantic and ontological knowledge into the process of Web usage mining, and particularly, in its application to Web personalization and recommender systems.

Domain knowledge can be integrated into the Web mining process in many ways. This includes leveraging explicit domain ontologies or implicit domain semantics extracted from the content or the structure of documents or Web site. In general, however, this process may involve one or more of three critical activities: domain ontology acquisition, knowledge base construction, and knowledge-enhanced pattern discovery.

Domain Ontology Acquisition

The process of acquiring, maintaining and enriching the domain ontologies is referred to as "ontology engineering". For small Web sites with only static Web pages, it is feasible to construct a domain knowledge base manually or semi-manually. In Loh, Wives, & de Oliveira (2000) a semi-manual approach is adopted for defining each domain concept as a vector of terms with the help of existing vocabulary and natural language processing tools.

However, manual construction and maintenance of domain ontologies require a great deal of effort on the part of knowledge engineers, particularly for large-scale Web sites or Web sites with dynamically generated content. In dynamically generated Web sites, page templates are usually populated based on structured queries performed against back-end databases. In such cases, the database schema can be used directly to acquire ontological information. Some Web servers send structured data files (e.g., XML files) to users and let client-side formatting mechanisms (e.g., CSS files) work out the final Web representation on client agents. In this case, it is generally possible to infer the schema from the structured data files.

When there is no direct source for acquiring domain ontologies, machine learning and text mining techniques must be employed to extract domain knowledge from the content or hyperlink structure of the Web pages. In Clerkin, Cunningham, & Hayes (2001) a hierarchical clustering algorithm is applied to terms in order to create concept hierarchies. In Stumme, Taouil, Bastide, Pasquier, & Lakhal (2000) a Formal Concept Analysis framework is proposed to derive a concept lattice (a variation of association rule algorithm). The approach proposed in Maedche & Staab (2000) learns generalized conceptual relations by applying association rule mining. All these efforts aim to **automatically** generate machine understandable ontologies for Web site domains.

The outcome of this phase is a set of formally defined domain ontologies that precisely represent the Web site. Good representation should provide machine understandability, the power of reasoning, and computation efficiency. The choice of ontology representation language has a direct effect on the flexibility of the data mining phase. Common representation approaches are vector-space model (Loh et al., 2000), descriptive logics (such as DAML+OIL)

(Giugno & Lukasiewicz, 2002; Horrocks & Sattler 2001), first order logic (Craven et al., 2000), relational models (Dai & Mobasher, 2002), probabilistic relational models (Getoor, Friedman, Koller, & Taskar, 2001), and probabilistic Markov models (Anderson, Domingos, & Weld, 2002).

Knowledge Base Construction

The first phase generates the formal representation of concepts and relations among them. The second phase, knowledge base construction, can be viewed as building mappings between concepts or relations on the one hand, and objects on the Web. The goal of this phase is to find the instances of the concepts and relations from the Web site's domain, so that they can be exploited to perform further data mining tasks. Learning algorithms play an important role in this phase.

In Ghani & Fano (2002) a text classifier is learned for each "semantic feature" (somewhat equivalent to the notion of a concept) based on a small manually labeled data set. First Web pages are extracted from different Web sites that belong to a similar domain, and then the semantic features are manually labeled. This small labeled data set is fed into a learning algorithm as the training data to learn the mappings between Web objects and the concept labels. In fact, this approach treats the process of assigning concept labels as filling "missing" data. Craven et al. (2000) adopt a combined approach of statistical text classification and first order text classification in recognizing concept instances. In that study, learning process is based on both page content and linkage information.

Knowledge-Enhanced Web Data Mining

Domain knowledge enables analysts to perform more powerful Web data mining tasks. The applications include content mining, information retrieval and extraction, Web usage mining, and personalization. On the other hand, data mining tasks can also help to enhance the process of domain knowledge discovery.

Domain knowledge can improve the accuracy of document clustering and classification and induce more powerful content patterns. For example, in Horrocks (2002), domain ontologies are employed in selecting textual features. The selection is based on lexical analysis tools that map terms into concepts within the ontology. The approach also aggregates concepts by merging the concepts that have low support in the documents. After preprocessing, only necessary concepts are selected for the content clustering step. In McCallum, Rosenfeld, Mitchell, & Ng (1998) a concept hierarchy is used to improve the accuracy and the scalability of text classification.

Traditional approaches to content mining and information retrieval treat every document as a set or a bag of terms. Without domain semantics, we would treat “human” and “mankind” as different terms, or, “brake” and “car” as unrelated terms. In Loh et al. (2000) a concept is defined as a group of terms that are semantically relevant, e.g., as synonyms. With such concept definitions, concept distribution among documents is analyzed to find interesting concept patterns. For example, one can discover dominant themes in a document collection or in a single document; or find associations among concepts.

Ontologies and domain semantics have been applied extensively in the context of Web information retrieval and extraction. For example, the ARCH system (Parent, Mobasher, & Lytinen, 2001) adopts concept hierarchies because they allow users to formulate more expressive and less ambiguous queries when compared to simple keyword-based queries. In ARCH, an initial user query is used to find matching concepts within a portion of concept hierarchy. The concept hierarchy is stored in an aggregated form with each node represented as a term vector. The user can select or unselect nodes in the presented portion of the hierarchy, and relevance feedback techniques are used to modify the initial query based on these nodes.

Similarly, domain-specific search and retrieval applications allow for more focused and accurate search based on specific relations inherent in the underlying domain knowledge. The CiteSeer system (Bollacker, Lawrence, & Giles, 1998) is a Web agent for finding interesting research publications, in which the relation “cited by” is the primary relation discovered among objects (i.e., among published papers). Thus, CiteSeer allows for comparison and retrieval of documents, not only based on similar content, but also based on inter-citation linkage structure among documents.

CiteSeer is an example of an approach for integrating semantic knowledge based on Web structure mining. In general, Web structure mining tasks take as input the hyperlink structure of Web pages (either belonging to a Web site or relative to the whole Web), and output the underlying patterns (e.g., page authority values, linkage similarity, Web communities, etc.) that can be inferred from the hypertext co-citations. Another example of such an approach is the PageRank algorithm which is the backbone of the Google search engine. PageRank uses the in-degree of indexed pages (i.e., number of pages referring to it) in order to rank pages based on quality or authoritativeness. Such algorithms that are based on the analysis of structural attributes, can be further enhanced by integrating content semantics (Chakrabarti et al., 1998). Web semantics can also enhance crawling algorithms by combining content or ontology with linkage information (Chakrabarti, van den Berg, & Dom, 1999; Maedche, Ehrig, Handschuh, Volz, & Stojanovic, 2002).

The use of domain knowledge can also provide tremendous advantages in Web usage mining and personalization. For example, semantic knowledge may help in interpreting, analyzing, and reasoning about usage patterns discovered in the mining phase. Furthermore, it can enhance collaborative filtering and personalization systems by providing concept-level recommendations (in contrast to item-based or user-based recommendations). Another advantage is that user demographic data, represented as part of a domain ontology, can be more systematically integrated into collaborative or usage-based recommendation engines. Several studies have considered various approaches to integrate content-based semantic knowledge into traditional collaborative filtering and personalization frameworks (Anderson et al., 2002; Claypool et al., 1999; Melville, Mooney, & Nagarajan, 2002; Mobasher, Dai, Luo, Sun, & Zhu, 2000b; Pazzani, 1999). Recently, we proposed a formal framework for integrating full domain ontologies with the personalization process based on Web usage mining (Dai & Mobasher, 2002).

Web Usage Mining and Personalization

The goal of personalization based on Web usage mining is to recommend a set of objects to the current (active) user, possibly consisting of links, ads, text, products, etc., tailored to the user's perceived preferences as determined by the matching usage patterns. This task is accomplished by matching the active user session (possibly in conjunction with previously stored profiles for that user) with the usage patterns discovered through Web usage mining. We call the usage patterns used in this context *aggregate usage profiles* since they provide an aggregate representation of the common activities or interests of groups of users. This process is performed by the recommendation engine which is the online component of the personalization system. If the data collection procedures in the system include the capability to track users across visits, then the recommendations can represent a longer term view of user's potential interests based on the user's activity history within the site. If, on the other hand, aggregate profiles are derived only from user sessions (single visits) contained in log files, then the recommendations provide a "short-term" view of user's navigational interests. These recommended objects are added to the last page in the active session accessed by the user before that page is sent to the browser.

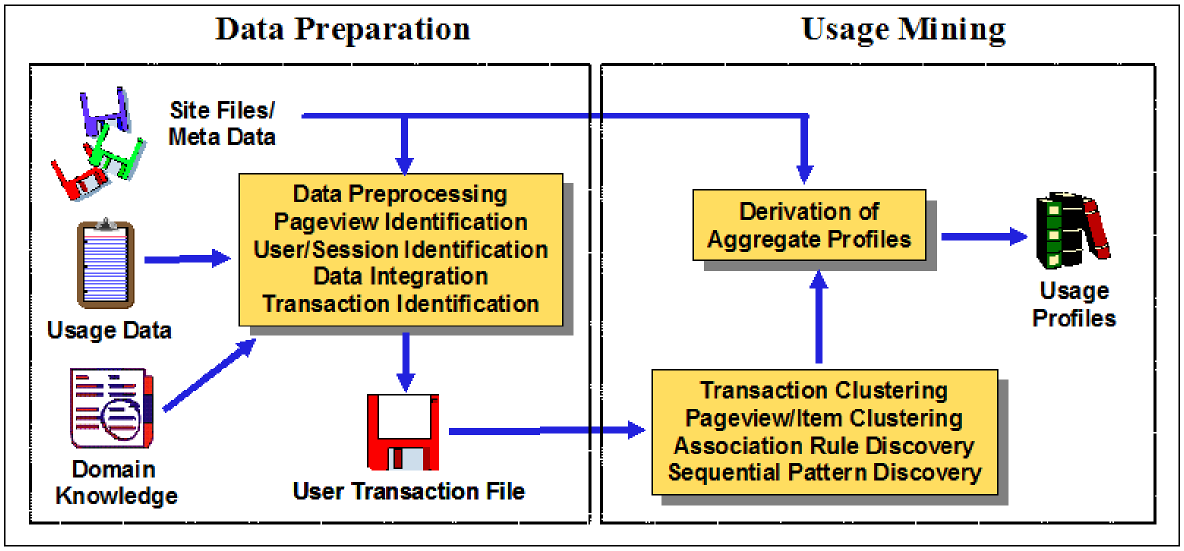


Figure 1: General Framework for Web Personalization Based on Web Usage Mining – The Offline Pattern Discovery Component.

The overall process of Web personalization based on Web usage mining consists of three phases: data preparation and transformation, pattern discovery, and recommendation. Of these, only the latter phase is performed in real-time. The data preparation phase transforms raw Web log files into transaction data that can be processed by data mining tasks. A variety of data mining techniques can be applied to this transaction data in the pattern discovery phase, such as clustering, association rule mining, and sequential pattern discovery. The results of the mining phase are transformed into aggregate usage profiles, suitable for use in the recommendation phase. The recommendation engine considers the active user session in conjunction with the discovered patterns to provide personalized content.

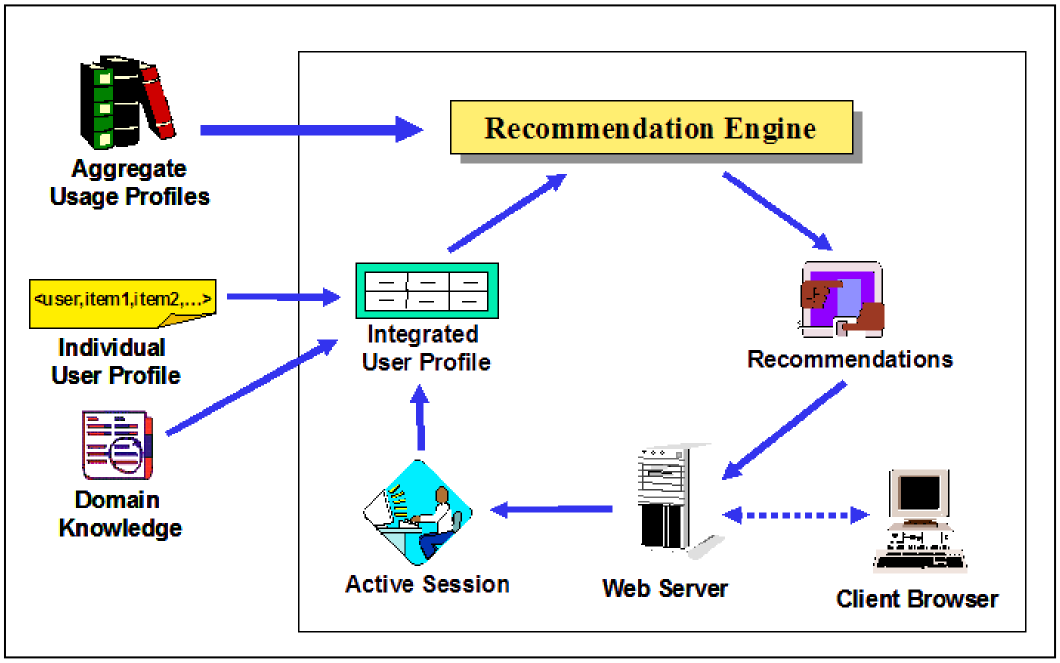


Figure 2: General Framework for Web Personalization Based on Web Usage Mining – The Online Personalization Component.

The primary data sources used in Web usage mining are the server log files, which include Web server access logs and application server logs. Additional data sources that are also essential for both data preparation and pattern discovery include the site files (HTML, XML, etc.) and meta-data, operational databases, and domain knowledge. Generally speaking, the data obtained through these sources can be categorized into four groups (see also Cooley, Mobasher, & Srivastava, 1999 and Srivastava et al., 2000).

- **Usage data**: The log data collected automatically by the Web and application servers represents the fine-grained navigational behavior of visitors. Depending on the goals of the analysis, this data needs to be transformed and aggregated at different levels of abstraction. In

Web usage mining, the most basic level of data abstraction is that of a *pageview.* Physically, a pageview is an aggregate representation of a collection of Web objects contributing to the display on a user's browser resulting from a single user action (such as a clickthrough). These Web objects may include multiple pages (such as in a frame-based site), images, embedded components, or script and database queries that populate portions of the displayed page (in dynamically generated sites). Conceptually, each pageview represents a specific "type" of user activity on the site, e.g., reading a news article, browsing the results of a search query, viewing a product page, adding a product to the shopping cart, and so on. On the other hand, at the user level, the most basic level of behavioral abstraction is that of a *server session* (or simply a *session).* A session (also commonly referred to as a "visit") is a sequence of pageviews by a single user during a single visit. The notion of a session can be further abstracted by selecting a subset of pageviews in the session that are significant or relevant for the analysis tasks at hand. We shall refer to such a semantically meaningful subset of pageviews as a *transaction.* It is important to note that a transaction does not refer simply to product purchases, but it can include a variety of types of user actions as captured by different pageviews in a session.

- **Content data**: The content data in a site is the collection of objects and relationships that are conveyed to the user. For the most part, this data is comprised of combinations of textual material and images. The data sources used to deliver or generate this data include static HTML/XML pages, image, video, and sound files, dynamically generated page segments from scripts or other applications, and collections of records from the operational database(s). The site content data also includes semantic or structural meta-data embedded within the site or individual pages, such as descriptive keywords, document attributes, semantic tags, or HTTP variables. Finally, the underlying domain ontology for the site is also considered part of content data. The domain ontology may be captured implicitly within the site, or it may exist in some explicit form. The explicit representation of domain ontologies may include conceptual hierarchies over page contents, such as product categories, structural hierarchies represented by the underlying file and directory structure in which the site content is stored, explicit representation of semantic content and relationships via an ontology language such as RDF, or a database schema over the data contained in the operational databases.

- Structure data: The structure data represents the designer's view of the content organization within the site. This organization is captured via the inter-page linkage structure among pages, as reflected through hyperlinks. The structure data also includes the intra-page structure of the content represented in the arrangement of HTML or XML tags within a page. For example, both HTML and XML documents can be represented as tree structures over the space of tags in the page. The structure data for a site is normally captured by an automatically generated "site map" which represents the hyperlink structure of the site. A site mapping tool must have the capability to capture and represent the inter- and intra-pageview relationships. This necessity becomes most evident in a frame-based site where portions of distinct pageviews may represent the same physical page. For dynamically generated pages, the site mapping tools must either incorporate intrinsic knowledge of the underlying applications and scripts, or must have the ability to generate content segments using a sampling of parameters passed to such applications or scripts.

- **User data**: The operational database(s) for the site may include additional user profile information. Such data may include demographic or other identifying information on registered users, user ratings on various objects such as pages, products, or movies, past purchase or visit histories of users, as well as other explicit or implicit representation of users' interests. Obviously, capturing such data would require explicit interactions with the users of the site. Some of this data can be captured anonymously, without any identifying user information, so long as there is the ability to distinguish among different users. For example, anonymous information contained in client-side cookies can be considered a part of the user's profile information, and can be used to identify repeat visitors to a site. Many personalization applications require the storage of prior user profile information. For example, collaborative filtering applications, generally, store prior ratings of objects by users, though, such information can be obtained anonymously, as well.

For a detailed discussion of preprocessing issues related to Web usage mining see Cooley et al., 1999. Usage preprocessing results in a set of n pageviews, $P = \{p_1, p_2, \cdots, p_n\}$, and a set of m user transactions, $T = \{t_1, t_2, \cdots, t_m\}$, where each t_i in T is a subset of P. *Pageviews* are semantically meaningful entities to which mining tasks are applied (such as pages or products). Conceptually, we view each transaction *t* as an *l*-length sequence of ordered pairs:

$$t = \left\langle (p_1^t, w(p_1^t)), (p_2^t, w(p_2^t)), \cdots, (p_l^t, w(p_l^t)) \right\rangle,$$

where each $p_i^t = p_j$ for some *j* in $\{1, \cdots, n\}$, and $w(p_i^t)$ is the weight associated with pageview p_i^t in transaction *t*, representing its significance (usually, but not exclusively, based on time duration).

For many data mining tasks, such as clustering and association rule discovery, as well as collaborative filtering based on the *k*NN technique, we can represent each user transaction as a vector over the *n*-dimensional space of pageviews. Given the transaction *t* above, the transaction vector $\vec{t}$ is given by: $t = \left\langle w_{p_1}^t, w_{p_2}^t, \cdots, w_{p_n}^t \right\rangle$, where each $w_{p_i}^t = w(p_j^t)$, for some *j* in $\{1, \cdots, n\}$, if p_j appears in the transaction *t*, and $w_{p_i}^t = 0$, otherwise. Thus, conceptually, the set of all user transactions can be viewed as an $m \times n$ transaction-pageview matrix, denoted by *TP*.

Given a set of transactions as described above, a variety of unsupervised knowledge discovery techniques can be applied to obtain patterns. These techniques such as clustering of transactions (or sessions) can lead to the discovery of important user or visitor segments. Other techniques such as item (e.g., pageview) clustering and association or sequential pattern discovery can be used to find important relationships among items based on the navigational patterns of users in the site. In each case, the discovered patterns can be used in conjunction with the active user session to provide personalized content. This task is performed by a recommendation engine.

REQUIREMENTS FOR SEMANTIC WEB USAGE MINING

In this section, we present and discuss the essential requirements in the integration of domain knowledge with Web usage data for pattern discovery. Our focus is on the critical tasks that particularly play an important role when the discovered patterns are to be used for Web personalization. As a concrete example, in the last part of this section we discuss an approach for integrating semantic features extracted from the content of Web sites with Web usage data, and how this integrated data can be used in conjunction with clustering to perform personalization. In the next section, we go beyond keyword-based semantics and present a more formal framework for integrating full ontologies with the Web usage mining and personalization processes.

Representation of Domain Knowledge

Representing Domain Knowledge as Content Features

One direct source of semantic knowledge that can be integrated into mining and personalization processes is the textual content of Web site pages. The semantics of a Web site are, in part, represented by the content features associated with items or objects on the Web site. These features include keywords, phrases, category names, or other textual content embedded as meta-information. Content preprocessing involves the extraction of relevant features from text and meta-data.

During the preprocessing, usually different weights are associated with features. For features extracted from meta-data, feature weights are usually provided as part of the domain knowledge specified by the analyst. Such weights may reflect the relative importance of certain concepts. For features extracted from text, weights can normally be derived automatically, for example as a function of the term frequency and inverse document frequency (tf.idf) which is commonly used in information retrieval.

Further preprocessing on content features can be performed by applying text mining techniques. This would provide the ability to filter the input to, or the output from, other mining algorithms. For example, classification of content features based on a concept hierarchy can be used to limit the discovered patterns from Web usage mining to those containing pageviews about a certain subject or class of products. Similarly, performing learning algorithms such as, clustering, formal concept analysis, or association rule mining on the feature space can lead to composite features representing concept categories or hierarchies (Clerkin, Cunningham, & Hayes, 2001; Stumme et al., 2000).

The integration of content features with usage-based personalization is desirable when we are dealing with sites where text descriptions are dominant and other structural relationships in the data are not easy to obtain, e.g., news sites or online help systems, etc. This approach, however, is incapable of capturing more complex relations among objects at a deeper semantic level based on the inherent properties associated with these objects. To be able to recommend different types of complex objects using their underlying properties and attributes, the system must be able to rely on the characterization of user segments and objects, not just based on keywords, but at a deeper semantic level using the domain ontologies for the objects. We will discuss some examples of how integrated content features and usage data can be used for personalization later in this Section.

Representing Domain Knowledge as Structured Data

In Web usage mining, we are interested in the semantics underlying a Web transaction or a user profile which is usually composed of a group of pageview names and query strings (extracted from Web server logs). Such features, in isolation, do not convey the semantics associated with the underlying application. Thus, it is important to create a mapping between these features and the objects, concepts, or events they represent.

Many E-Commerce sites generate Web pages by querying operational databases or semi-structured data (e.g., XML and DTDs), from which semantic information can be easily derived. For Web sites in which such structured data cannot be easily acquired, we can adopt machine learning techniques to extract semantic information [10]. Furthermore, the domain knowledge acquired should be machine understandable in order to allow for further processing or reasoning. Therefore, the extracted knowledge should be represented in some standard knowledge representation language.

DAML+OIL (Horrocks & Sattler, 2001) is an example of an ontology language that combines the Web standards from XML and RDF, with the reasoning capabilities from a description logic *SHIP(DL).* The combinations of relational models and probabilistic models is another common approach to enhance Web personalization with domain knowledge and reasoning mechanism. Several approaches to personalization have used Relational Models such as Relational Markov Model (Anderson et al., 2002). Both of these approaches provide the ability to represent knowledge at different levels of abstraction, and the ability to reason about concepts, including about such relations as subsumption and membership.

In Dai & Mobasher (2002), we adopted the syntax and semantics of another ontology representation framework, *SHOQ(D),* to represent domain ontologies. In *SHOQ(D),* the notion of *concrete datatype* is used to specify literal values and *individuals* which represent real objects in the domain ontology. Moreover, *concepts* can be viewed as sets of individuals, and *roles* are binary relations between a pair of concepts or between concepts and data types. The detailed formal definitions for concepts and roles are given in Horrocks & Sattler (2001) and Giugno & Lukasiewicz (2002). Because our current work does not focus on reasoning tasks such as deciding subsumption and membership, we do not focus our discussion on these operations. The reasoning apparatus in *SHOQ(D)* can be used to provide more intelligent data mining services as part of our future work.

Building "Mappings" Between Usage-Level and Domain-Level Instances

During usage data preprocessing or post processing, we may want to assign domain semantics to user navigational paths/patterns by mapping the pageview names or URLs (or queries) to the instances in the knowledge base. To be more specific, instead of describing a user's navigational path as: "a_1, a_2, ..., a_n" (where a_i is a URL pointing to a Web resource), we need to represent it using the instances from the knowledge base, such as: "movie(name=Matrix), movie(name=Spiderman), …, movie(name=Xman)." With the help of a pre-acquired concept hierarchy, we may, for example, be able to infer that the current user's interest is in the category of "Action&Sci-Fi." We refer to this "semantic" form of usage data as "Semantic User Profiles." These profiles, in turn, can be used for semantic pattern discovery and online recommendations. In the context of personalization applications, domain-level (semantic) instances may also need to be mapped back to Web resources or pages. For example, a recommendation engine using semantic user profiles may result in recommendations in the form of a movie genre. This concept must be mapped back into specific pages, URLs, or sections of the site relating to this genre before recommendations can be relayed to the user.

Using Content and Structural Characteristics

Classification algorithms utilizing content and structural features from pages are well-suited for creating mappings from usage data to domain-level instances. For example, in Craven et al. (2000) and Ghani & Fano (2002) classifiers are trained that exploit content or structural features (such as terms, linkage information, and term proximity) of the pageviews. From the pageview names or URLs we can obtain the corresponding Web content such as meta-data or keywords. With help from text classification algorithms, it is possible to efficiently map from keywords to attribute instances (Ghani & Fano, 2002).

Another good heuristics used in creating semantic mappings is based on the anchor text associated with hyperlinks. If we can build the complete user navigational path, we would be able to acquire the anchor text for each URL or pageview name. We can include the anchor text as part of the content features extracted from the body of documents or in isolation. However, whereas the text features in a document represent the semantics of the document, itself, the anchor text represents the semantics of the document to which the associated hyperlink points.

Using Query Strings

So far, we have overlooked the enormous amount of information stored in databases or semi-structured documents associated with a site. Large information servers often serve content integrated from **multiple** underlying servers and databases (Berendt, & Spiliopoulou, 2000). The dynamically generated pages on such servers are based on queries with multiple parameters attached to the URL corresponding to the underlying scripts or applications. Using the Web server query string recorded in the server log files it is possible to reconstruct the response pages. For example, the following are query strings from a hypothetical online book-seller Web site:

```
http://www.xyz.com/app.cgi?action=viewitem&item=1234567&category=1234
http://www.xyz.com/app.cgi?action=search&searchtype=title&searchstring=web+mining
http://www.xyz.com/app.cgi?action=order&item=1234567&category=1234&couponid=3456
```

If the background database or semi-structured documents are available, then we can access the content of the instances in the response pages via the name-value pairs from the query strings. This enriches our knowledge base of user interest. In the above book-seller Web site example, if we were able to access background database, we would be able to get the content of item "1234567" in category "1234". In this case, we could have the book name, price, author information of this item. We could recommend other books in the same content category or written by the same author. More generally, in well-designed sites, there is usually an explicitly available semantic mapping between query parameters and objects (such as products and categories), which would obviate the need to reconstruct the content of dynamic pages.

Levels of Abstraction

Capturing semantic knowledge at different levels of abstraction provides more flexibility both in the mining phase and in the recommendation phase. For example, focusing on higher-level concepts in a concept hierarchy would allow certain patterns to emerge which otherwise may be missed due to low support. On the other hand, the ability to drill-down into the discovered patterns based on finer-grained subconcepts would provide the ability to give more focused and useful recommendations.

Domain knowledge with attributes and relations requires the management of a great deal more data than is necessary in traditional approaches to Web usage mining. Thus, it becomes essential to prune unnecessary attributes or relations. For example, it may be possible to examine the number of distinct values of each attribute and generalize the attributes if there is a concept hierarchy over the attribute values. In Han & Fu (1995) a multiple-level association rule mining algorithm is proposed that utilizes concept hierarchies. For example, the usage data in our hypothetical movie site may not provide enough support for an association rule: "Spiderman, Xmen ~ Xmen2", but mining at a higher level may result in obtaining a rule: "Sci-Fi&Action,

Xmen ~ Xmen2”. In Anderson et al. (2002) relational Markov models are built by performing shrinkage (McCallum et al., 1998) between the estimates of parameters at all levels of abstractions relative to a concept hierarchy. If a pre-specified concept hierarchy does not exist, it is possible to automatically create such hierarchies through a variety of machine learning techniques, such as hierarchical agglomerative clustering (Stumme et al., 2000).

Integration of Semantics at Different Stages of Knowledge Discovery

The semantic information stored in the knowledgebase can be leveraged at various steps in the knowledge discovery process, namely in the preprocessing phase, in the pattern discovery phase, or during the post-processing of the discovered patterns.

Preprocessing phase

The main task of data preprocessing is to prune noisy and irrelevant data, and to reduce data volume for the pattern discovery phase. In Mobasher, Dai, Luo, & Nakagawa (2002), it was shown that applying appropriate data preprocessing techniques on usage data could improve the effectiveness of Web personalization. The concept level mappings from the pageview-level data to concepts can also be performed in this phase. This results in a transformed transaction data to which various data mining algorithms can be applied. Specifically, the transaction vector *t* given previously can be transformed into a vector $t' = \left\langle w^t_{o_1}, w^t_{o_2}, \cdots, w^t_{o_k} \right\rangle$, where each o_j is a semantic object appearing in one of the pageviews contained in the transaction, and $w^t_{o_j}$ is a weight associated with that object in the transaction. These semantic objects may be concepts appearing in the concept hierarchy or finer-grained objects representing instances of these concepts.

Pattern discovery phase

Successful utilization of domain knowledge in this phase requires extending basic data mining algorithms to deal with relational data to concept hierarchies. As an example, consider a distance-based data mining technique such as clustering. The clustering of flat single-relation data (such as Web user transactions) involves the computation of similarities or distance among transaction vectors. In such cases, normally simple vector-based operations are used. However, in the presence of integrated domain knowledge represented as concept hierarchies or ontologies, the clustering algorithms will have to perform much more complex similarity computations across dimensions and attributes. For example, even if the two user transactions have no pageviews in common, they may still be considered similar provided that the items occurring in both transactions are themselves "similar" based on some of their attributes or properties. The integration of domain knowledge will generate "semantic" usage patterns, introducing great flexibility as well as challenges. The flexibility lies in the pattern discovery being independent of item identities. The challenge is in the development of scalable and efficient algorithms to perform the underlying computational tasks such as similarity computations. We discuss this issue further below.

Post-processing phase

Exploiting domain knowledge in this phase can be used to further explain usage patterns or to filter out irrelevant patterns. One possibility is to first perform traditional usage mining tasks on the item-level usage data obtained in the preprocessing phase, and then use domain knowledge to interpret or transform the item level user profiles into "domain-level usage profiles" (Mobasher & Dai, 2002) involving concepts and relations in the ontology. The advantage of this approach is that we can avoid the scalability issues that can be endemic in the pattern discovery phase. The disadvantage is that some important structural relationships may not be used during the mining phase resulting in lower quality patterns.

Aggregation Methods for Complex Objects

To characterize patterns discovered through data mining techniques, it is usually necessary to derive aggregate representation of the patterns. An example of this situation is clustering applications. In the context of Web user transactions, clustering may result in a group of sessions or visitors that are considered similar because of their common navigational patterns. The vector representation of these transactions facilitates the aggregation tasks: the centroid (mean vector) of the transaction cluster acts as a representative of all of the transactions in that cluster. However, in the case of semantically enhanced transactions, the aggregation may have to be performed independently for each of the attributes associated with the objects contained in the cluster.

For example, clustering may result in a group of users who have all visited pages related to several movies. To be able to characterize this group of users at a deeper semantic level, it would be necessary to create an aggregate representation of the collection of movies in which they are interested. This task would require aggregation along each dimension corresponding to the attributes of "movie" instances, such as "genre", "actors", "directors", etc. Since each of these attributes require a different type of aggregation function depending on the data type and the domain, it may be necessary to associate various aggregation functions with the specification of the domain ontology, itself. In the next section we present one approach for solving this problem.

Measuring Semantic Similarities

Measuring similarities (alternatively, distances) among objects is a central task in many data mining algorithms. In the context of Web usage mining this may involve computing similarity measures among pageviews, among user transactions, or among users. This also becomes a critical task in personalization: a current user's profile must be matched with similar aggregate profiles representing the discovered user patterns or segments. As in the case of the aggregation problem discussed above, when dealing with semantically enhanced transactions, measuring similarities poses additional challenges. This is because the similarity of two transactions depends on the similarities of the semantic objects contained within the transactions.

Let us again consider the *static* vector model for representing a Web transaction t (or a user profile): $t = \left\langle w^t_{p_1}, w^t_{p_2}, \cdots, w^t_{p_n} \right\rangle$. Computing similarity between two such vectors is straightforward and can be performed using measures such as cosine similarity, Euclidean distance, Pearson correlation (e.g., in case the weights represent user ratings).

When such vectors are transformed according to the underlying semantics, however, the computation of similarities will involve the computation of semantic similarities among the concepts or objects, possibly using different domain-specific similarity measures. Let *A* and *B* be two transformed transactions, each represented as a set of semantic objects in a site:

$$A = \{a_1, a_2, \cdots, a_m\} \text{ and } B = \{b_1, b_2, \cdots, b_l\}.$$

The computation of vector similarity between *A* and *B*, *Sim*(*A*,*B*), is dependent on the semantic similarities among the component objects, *SemSim*(a_i,b_j). For instance, one approach might be to compute the weighted sum or average of the similarities among object pairs, such as in:

$$Sim(A,B) = \frac{\sum_{a \in A} \sum_{b \in B} SemSim(a,b)}{|A| \cdot |B|}$$

In general, computing the semantic similarity, *SemSim*(*a,b*), is domain dependent and requires knowledge of the underlying structure of among objects. If both objects can be represented using the same vector model (e.g., pages or documents represented as bags of words), we can compute their similarity using standard vector operations. On the other hand, if their representation includes attributes and relations specified in the domain ontology, we need to first make sure that the objects can be classified under a common ontological schema and then measure similarities along the different dimensions corresponding to each attribute. The notion of semantic matching among objects and classes has been a subject of considerable study recently (Rodriguez & Egenhofer, 2003; Palopoli, Sacca, Terracina, & Ursino, 2003; Ganesan, Garcia-Molina, & Widom, 2003).

For example, such an approach was used in Jin & Mobasher (2003) in the context of collaborative filtering with movies. In this work, association analysis was first performed on the "genre" attribute to define a genre hierarchy. Furthermore, the "year" attribute was discretized into intervals, while other attributes, such as "cast", were treated as a bag of words. These preprocessing steps allowed for the definition of appropriate similarity measures for each attribute. Finally, the semantic similarity between two movies, *i* and *j*, was defined as a linear combination of attribute-level similarities:

$$SemSim(i,j) = \alpha_1 * CastSim(i,j) + \alpha_2 * DirectorSim(i,j) + \alpha_3 * GenreSim(i,j) + \ldots,$$

where, α_i are predefined weights for the corresponding attributes.

Example: Using Content Features for Semantic Web Usage Mining

As an example of integrating semantic knowledge with the Web usage mining process, let us consider the especial case of using textual features from the content of Web pages to represent the underlying semantics for the site. As noted earlier, each pageview p can be represented as a k-dimensional feature vector, where k is the total number of extracted features (words or concepts) from the site in a global dictionary. This vector can be given by:

$$p = \langle fw(p, f_1), fw(p, f_2), \cdots, fw(p, f_k) \rangle$$

where $fw(p, f_j)$ is the weight of the jth feature in pageview p, for $1 \leq j \leq k$. For the whole collection of pageviews in the site, we then have an $n \times k$ pageview-feature matrix $PF = \{p_1, p_2, \cdots, p_n\}$.

There are now at least two basic choices as to when content features can be integrated into the usage-based personalization process: pre-mining integration or post-mining integration.

The pre-mining integration involves the transformation of user transactions, as described earlier, into "content-enhanced" transactions containing the semantic features of the pageviews. While, in practice, there are several ways to accomplish this transformation, the most direct approach involves mapping each pageview in a transaction to one or more content features. The range of this mapping can be the full feature space, or feature sets (composite features) which in turn may represent concepts and concept categories. Conceptually, the transformation can be viewed as the multiplication of the transaction-pageview matrix TP, defined earlier, with the pageview-feature matrix PF. The result is a new matrix $TF = \{t_1, t_2, \cdots, t_n\}$, where each t_i is a k-dimensional vector over the feature space. Thus, a user transaction can be represented as a content feature vector, reflecting that user's interests in particular concepts or topics.

Various data mining tasks can now be performed on the content-enhanced transaction data. For instance, if we apply association rule mining to such data, then we can get a group of association rules on content features. As an example, consider a site containing information about movies. This site may contain pages related to the movies themselves, actors appearing in the movies, directors, and genres. Association rule mining process could generate a rule such as: {"British", "Romance", "Comedy" $\Rightarrow$ "Hugh Grant"}, suggesting that users who are interested in British romantic comedies may also like the actor Hugh Grant (with a certain degree of confidence). During the online recommendation phase, the user's active session (which is also transformed into a feature vector) is compared with the discovered rules. Before recommendations are made, the matching patterns must be mapped back into Web pages or Web objects. In the above example, if the active session matches the left hand side of the association rule, the recommendation engine could recommend other Web pages that contain the feature "Hugh Grant".

The post-mining integration of semantic features involves combining the results of mining (performed independently on usage and content data) during the online recommendation phase. An example of this approach was presented in Mobasher et al. (2000b), where clustering algorithms were applied to both the transaction matrix *TP* and the transpose of the feature matrix *PF.* Since both matrices have pageviews as dimensions, the centroids of the resulting clusters in both cases can be represented as sets of pageview-weight pairs where the weights signify the frequency of the pageview occurrence in the corresponding cluster. We call the patterns generated from content data "content profiles", while the patterns derived from usage data are called "usage profiles". Though they share the same representation, they have different semantics: usage profiles represent a set of transactions with similar navigational behavior, while content profiles contain groups of Web pages with (partly) similar content.

Specifically, given a transaction cluster (respectively, a feature cluster) *cl,* we can construct the usage (respectively, content) profile pr_{cl} as a set of pageview-weight pairs by computing the centroid of *cl:*

$$pr_{cl} = \{\langle p, weight(p, pr_{cl})\rangle \mid weight(p, pr_{cl}) \geq \mu\},$$

where:

- the significance weight, $weight(p, pr_{cl})$, of the page p within the usage (respectively, content) profile pr_{cl} is given by

$$weight(p, pr_{cl}) = \frac{1}{|cl|} \cdot \sum_{s \in cl} w(p,s)$$

- $w(p,s)$ is the weight of page p in transaction (respectively, feature) vector s in the cluster cl; and
- the threshold μ is used to focus only on those pages in the cluster that appear in a sufficient number of vectors in that cluster.

Each such profile, in turn, can be represented as a vector in the original n-dimensional space of pageviews. This aggregate representation can be used directly in the recommendation phase: given a new user, u who has accessed a set of pages, P_u, so far, we can measure the similarity of P_u to the discovered profiles, and recommend to the user those pages in matching profiles which have not yet been accessed by the user. Note that this approach does not distinguish between recommendations emanating from the matching content and usage profiles. Also note that there are many other ways of combining usage profiles and content profiles during the online recommendation phase. For example, we can use content profiles as the last resort in the situation where usage profiles can not provide sufficient number of recommendations.

A FRAMEWORK FOR ONTOLOGY-BASED PERSONALIZATION

At a conceptual level, there may be many different kinds of objects within a given site that are accessible to users. At the physical level, these objects may be represented by one or more Web pages. For example, our hypothetical movie site may contain pages related to the movies, actors, directors, and studios. Conceptually, each of these entities represents a different type of semantic object. During a visit to this site, a user may implicitly access several of these objects together during a session by navigating to various pages containing them. In contrast to content features, ontological representation of domain knowledge contained in the site makes it possible to have a uniform architecture to model such objects, their properties, and their relationships. Furthermore, such a representation would allow for a more natural mapping between the relational schema for the backend databases driving Web applications and the navigational behavior of users.

In this section we will present a general framework for utilizing domain ontologies in Web usage mining and personalization. Figure 3 lays out a general process for such an integrated approach. In keeping with our earlier discussion, it is composed of 3 main phases: preprocessing, pattern discovery and online recommendation. Each of these phases must take into account the object properties and their relationships as specified in a domain ontology.

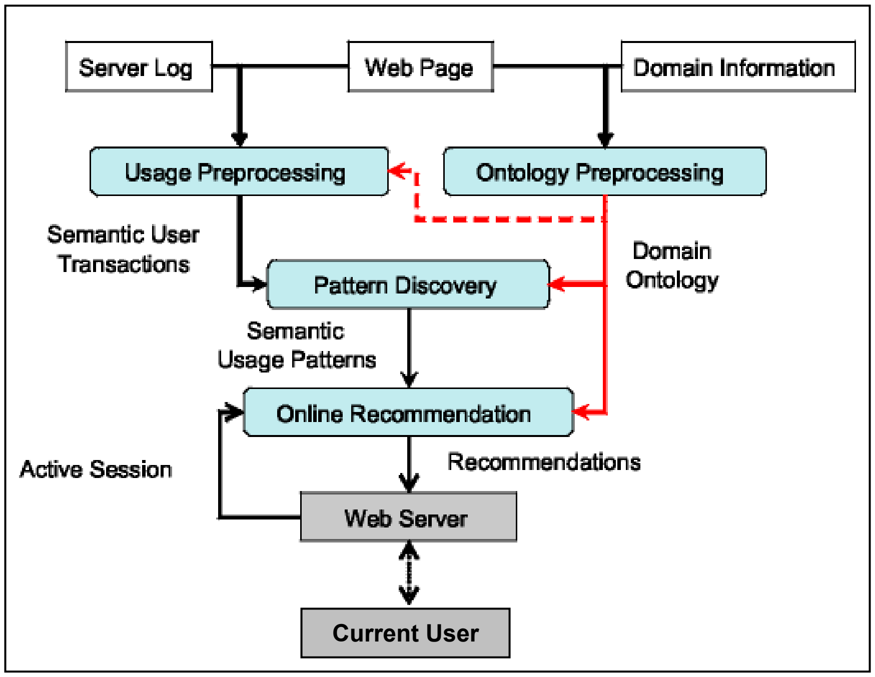


Figure 3: A General Framework for Personalization Based on Domain Ontologies

We assume that the site ontology is already available (either specified manually, or extracted automatically using ontology learning techniques). The goal of the preprocessing phase is to transform users' navigational transactions into "semantic transaction" by mapping accessed pages and resource to concepts and objects of the specified ontology. The goal of the pattern discovery phase is to create aggregate representation of groups of semantic objects that are implicitly accessed by similar users, thus providing a semantic characterization of user segments with common behavior or interests. Finally, in the recommendation phase, the discovered semantic patterns are utilized in conjunction with an ongoing record of a current user's activity (including, possibly the user's stored profile) to recommend new resources, pages, or objects to that user.

Knowledge Representation

General ontology representation languages such as DAML+OIL (Horrocks, 2002) provide formal syntax and semantics for representing and reasoning with various elements of an ontology. These elements include "individuals" (or objects), "concepts" (which represent sets of individuals), and "roles" (which specify object properties). In DAML+OIL*,* the notion of a concept is quite general and may encompass a heterogeneous set of objects with different properties (roles) and structures. We, on the other hand, are mainly interested in the aggregate representations for groups of objects that have a homogenous concept structure (i.e., have similar properties and data types). For example, we may be interested in a group of movie objects, each of which has specific values for properties such as "year", "genre", and "actors." We call such a group of objects a *class*. Thus, in our framework, the notion of a class represents a restriction of the notion of a concept in DAML+OIL. It should be noted, however, that the users of a Web site, in general, access a variety of objects belonging to different classes. Thus, this homogeneity assumption would imply that semantic objects within user transactions must first be classified into homogenous classes as a preprocessing step.

More specifically, we define a *class C* as a set of objects together with a set of *attributes*. These attributes together define the internal properties of the objects in *C* or relationships with other concepts that involve the objects in *C*. Thus attributes of a class correspond to a subset of the set of roles in the domain ontology. We denote the domain of values of an attribute *r* as D_r. Furthermore, because we are specifically interested in aggregating objects at the attribute level, we extend the notion of a role to include a domain-specific combination function and an ordering relation.

More formally, a class *C* is characterized by a finite set of attributes *AC,* where each attribute *a* in *AC* is defined as follows.

Definition: Let *C* be a class in the domain ontology. An *attribute* $a \in AC$ is a 4-tuple $a = \langle T_a, D_a, \prec_a, \psi_a \rangle$, where

- T_a is the *type* for the values for the attribute *a.*
- D_a is the domain of the values for *a;*
- $\prec_a$ is an ordering relation among the values in D_a*;* and
- ψ_a is a *combination function* for the attribute *a.*

The "type" of an attribute in the above definition may be a concrete datatype (such as "string" or "integer") or it may be a set of objects (individuals) belonging to another class.

In the context of data mining, comparing and aggregating values are essential tasks. Therefore, ordering relations among values are necessary properties for attributes. We associate an ordering relation $\prec_a$ with elements in D_a for each attribute *a.* The ordering relation $\prec_a$ can be null (if no ordering is specified in the domain of values), or it can define a partial or a total order among the domain values. For standard types such as values from a continuous range, we assume the usual ordering. In cases when an attribute *a* represents a concept hierarchy, the domain values of *a* are a set of labels, and $\prec_a$ is a partial order representing the "is-a" relation.

Furthermore, we associate a data mining operator, called the *combination function,* ψ_a*,* with each attribute *a.* This combination function defines an aggregation operation among the corresponding attribute values of a set of objects belonging to the same class. This function is essentially a generalization of the "mean" or "average" function applied to corresponding dimension values of a set of vectors when computing the centroid vector. In this context, we assume that the combination function is specified as part of the domain ontology for each attribute of a class. An interesting extension would be to automatically learn the combination function for each attribute based on a set of positive and negative examples.

Classes in the ontology define the structural and semantic properties of objects in the domain which are "instances" of that class. Specifically, each object *o* in the domain is also characterized by a set of attributes A_o corresponding to the attributes of a class in the ontology. In order to more precisely define the notion of an object as an instance of a class, we first define the notion of an instance of an attribute.

Definition: Given an attribute $a = \langle T_a, D_a, \prec_a, \psi_a \rangle$ and an attribute $b = \langle T_b, D_b, \prec_b, \psi_b \rangle$, *b* is an *instance* of *a,* if $D_b \subseteq D_a$, $T_b = T_a$, $\psi_b = \psi_a$, and $\prec_b$ is a restriction of $\prec_a$ to D_b. The attribute *b* is a *null instance* of *a,* if *Db* is empty.

Definition: Given a class *C* with attribute set $A_C = \{a_1^C, a_2^C, \cdots, a_n^C\}$, we say that an object *o* is *an instance of C,* if *o* has attributes $A_o = \{a_1^o, a_2^o, \cdots, a_n^o\}$ such that each a_i^o is a, possibly null, instance of a_1^C.

Based on the definitions of attribute and object instances, we can now provide a more formal representation of the combination function ψ_a. Let *C* be a class and $\{o_1, o_2, \cdots, o_m\}$ a set of object instances of *C.* Let $a \in A_C$ be an attribute of class *C.* The combination function ψ_a can be represented by:

$$\psi_a\left(\left\{\langle a_{o_1}, w_1 \rangle, \langle a_{o_2}, w_2 \rangle, \cdots, \langle a_{o_m}, w_m \rangle\right\}\right) = \langle a_{agg}, w_{agg} \rangle,$$

where each a_{o_i} belonging to object o_i is an instance of the attribute *a,* and each w_i is a weight associated with that attribute instance (representing the significance of that attribute relative to the other instances). Furthermore, a_{agg} is a *pseudo instance* of *a* meaning that it is an instance of *a* which does not belong to a real object in the underlying domain. The weight w_{agg} of a_{agg} is a function of w_1, w_2, ..., w_n.

Given a set of object instances, $\{o_1, o_2, \cdots, o_m\}$ of a class *C,* a *domain-level aggregate profile* for these instances is obtained by applying the combination function for each attribute in *C* to all of the corresponding attribute instances across all objects o_1, o_2, ..., o_n.

Ontology Preprocessing

The ontology preprocessing phase takes as input domain information (such as database schema and metadata, if any) as well as Web pages, and generates the site ontology. For simple Web sites, ontologies can be easily designed manually or derived semi-automatically from the site content. However, it is more desirable to have automatic ontology acquisition methods for a large

Web site, especially in Web sites with dynamically generated Web pages. E-commerce Web sites, for instance, usually have well-structured Web content, including predefined metadata or database schema. Therefore it is easier to build automatic ontology extraction mechanisms that are site-specific.

There have been a number of efforts dealing with the ontology learning problem (Clerkin et al., 2001; Craven et al., 2000; Maedche & Staab, 2000). A wide range of information, such as thesauri, content features, and database schema can help to identify ontologies. Many of these approaches have focused on extracting ontological information from the Web, in general. In Berendt et al. (2002) the notion of "Semantic Web Mining" was introduced, including a framework for the extraction of a concept hierarchy and the application of data mining techniques to find frequently occurring combinations of concepts.

An Example

As an example, let us revisit our hypothetical movie Web site. The Web site includes collections of pages containing information about movies, actors, directors, etc. A collection of pages describing a specific movies might include information such as the movie title, genre, starring actors, director, etc. An actor or director's information may include name, filmography (a set of movies), gender, nationality, etc. The portion of domain ontology for this site, as described, contains the classes **Movie, Actor** and **Director** (see Figure 4). The collection of Web pages in the site represent a group of embedded objects that are the instances of these classes.

In our example, the class **Movie** has attributes such as *Year, Actor* (representing the relation "acted by"), *Genre,* and *Director*. The *Actor* and *Director* attributes have values that are other objects in the ontology, specifically, object instances of classes **Actor** and **Director,** respectively. The attribute *Year* is an example of an attribute whose datatype is positive integers with the usual ordering. The attribute *Genre* has a concrete datatype whose domain values in D_{Genre} are a set of labels (e.g., "Romance" and "Comedy"). The ordering relation $\prec_{Genre}$ defines a partial order based on the "is-a" relation among subsets of these labels (resulting in a concept hierarchy of Genres, a portion of which is shown in Figure 4).

Figure 5 shows a **Movie** instance "About a Boy" and its related attributes and relations extracted from a Web page. The schema of the class **Movie** is shown at the bottom left portion of the figure. Here we treat the classes **Genre** and **Year** as attributes of the class **Movie**. The instances of the ontology are shown at the bottom right of the figure. The *Genre* attribute contains a partial order among labels representing a concept hierarchy of movie genres. We use a restriction of this partial order to represent the genre to which the **Movie** instance belongs. The diagram also shows a keyword bag containing the important keywords in that page.

An attribute a of an object o has a domain D_a. In cases when the attribute has unique a value for an object, D_a is a singleton. For example, consider an object instance of class **Movie,** "About a Boy" (see Figure 5). The attribute *Actor* contains two objects "H. Grant" and "T. Collette" that are instances of the class **Actor** (for the sake of presentation we use the actors' names to stand for the object instances of **Actor**). Therefore, D_{Actor} = {"H. Grant", "T. Collette"}. Also, a real object may have values for only some of the attributes. In this case the other attributes have empty domains. For instance, the attribute *Director* in the example has an empty domain and is thus not depicted in the figure. We may, optionally, associate a weight with each value in the attribute domain D_a (usually in the range [0,1]). This may be useful in capturing the relative importance of each attribute value.

For example, in a given movie the main actors should have higher weights than other actors in the cast. In our example, the object "H. Grant" has weight 0.6 and the object "Toni Collette" has weight 0.4. Unless otherwise specified, we assume that the weight associated with each attribute value is 1. In the object o shown in Figure 5, the domain for the attribute *Genre* is the set of labels {Genre-All, Action, Romance, Comedy, Romantic Comedy, Black Comedy, Kids & Family}. The ordering relation $\prec^{o}_{Genre}$ is a restriction of $\prec_{Genre}$ to the subset {Genre-All, Comedy, Romantic Comedy, Kids & Family}.

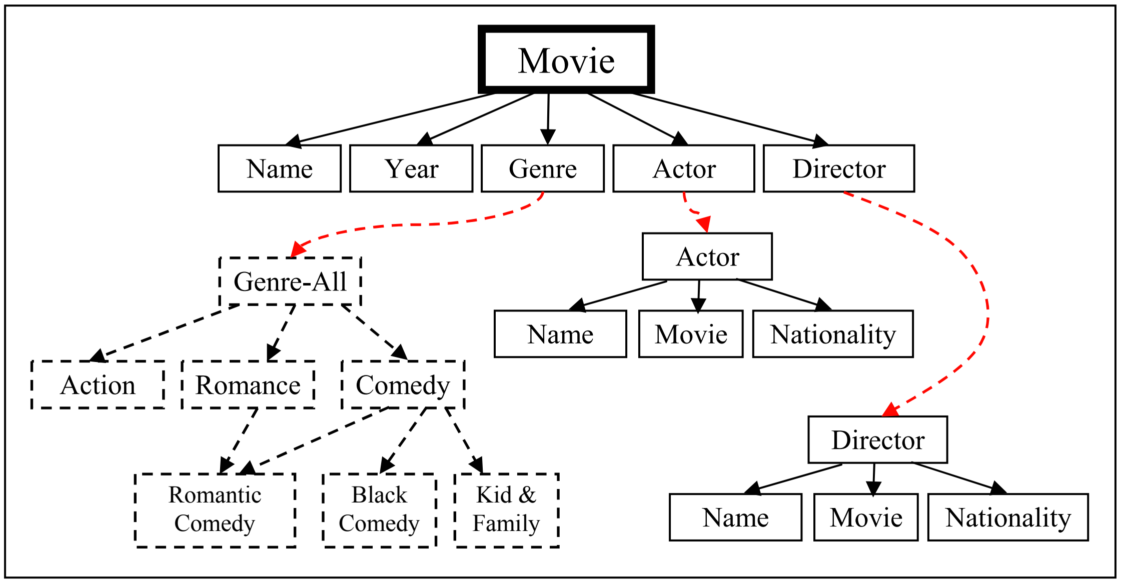
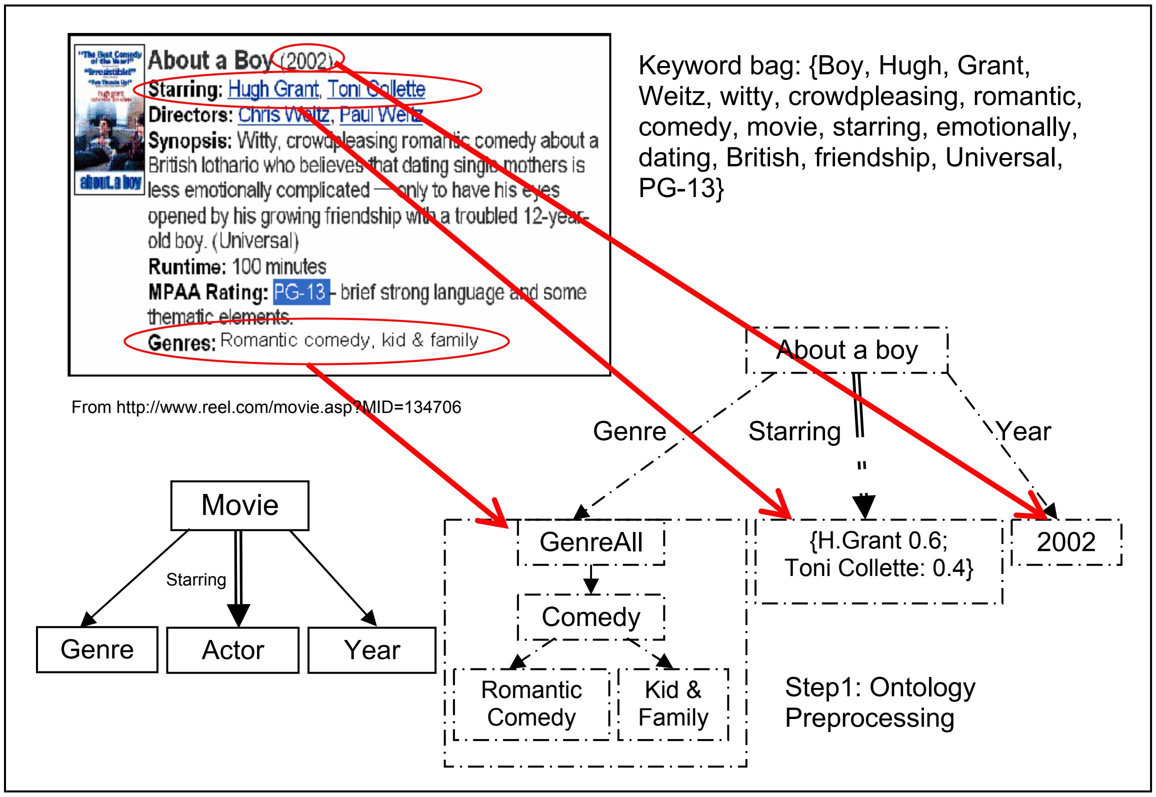


Figure 4: The Ontology for a movie Web site


About a Boy (2002)
Starring: Hugh Grant, Toni Collette
Directors: Chris Weitz, Paul Weitz
Synopsis: Witty, crowdpleasing romantic comedy about a British lothario who believes that dating single mothers is less emotionally complicated — only to have his eyes opened by his growing friendship with a troubled 12-year-old boy. (Universal)
Runtime: 100 minutes
MPAA Rating: PG-13 - brief strong language and some thematic elements.
Genres: Romantic comedy, kid & family

From http://www.reel.com/movie.asp?MID=134706

Keyword bag: {Boy, Hugh, Grant, Weitz, witty, crowdpleasing, romantic, comedy, movie, starring, emotionally, dating, British, friendship, Universal, PG-13}

Figure 5: Example of Ontology Preprocessing

Pattern Discovery

As depicted in Figure 3 domain ontologies can be incorporated into usage preprocessing to generate semantic user transactions, or they can be integrated into pattern discovery phase to generate semantic usage patterns. In the following example, we will focus on the latter approach.

Given a discovered usage profile (for example, a set of pageview-weight pairs obtained by clustering user transactions), we can transform it into a domain-level aggregate representation of the underlying objects (Dai & Mobasher, 2002). To distinguish between the representations we call the original discovered pattern an "item-level" usage profile, and we call the new profile based on the domain ontology a "domain-level" aggregate profile. The item-level profile is first represented as a weighted set of objects: $pr = \{\langle o_1, w_1 \rangle, \langle o_2, w_2 \rangle, \cdots, \langle o_n, w_n \rangle\}$ in which each o_i is an object in the underlying domain ontology and w_i represents o_i's significance in the profile *pr.* Here we assume that, either using manual rules, or through supervised learning methods, we can extract various object instances represented by the pages in the original page- or item-level usage profile. The transformed profile represents a set of objects accessed together frequently by a group of users (as determined through Web usage mining). Objects, in the usage profile, that belong to the same class are combined to form an aggregated pseudo object belonging to that class. An important benefit of aggregation is that the pattern volume is significantly reduced, thus relieving the computation burden for the recommendation engine. Our goal is to create an aggregate representation of this weighted set of objects to characterize the common interests of the user segment captured by the usage profile at the domain level.

Given the representation of a profile *pr* as a weighted set of objects, the objects in *pr* may be instances of different classes $C_1, C_2, ..., C_k$ in the ontology. The process of creating a domain-level aggregate profile begins by partitioning *pr* into collections of objects with each collection containing all objects that are instances of a specified class (in other words, the process of classifying the object instances in *pr).* Let G_i denote the elements of *pr* that are instances of the class C_i.

Having partitioned *pr* into *k* groups of homogeneous objects, $G_1, ..., G_k$, the problem is reduced to creating aggregate representation of each partition G_i. This task is accomplished with the help of the combination functions for each of the attributes of C_i some of whose object instances are contained in G_i. Once the representatives for every partition of objects are created, we assign a significance weight to each representative to mark the importance of this group of objects in the profile. In our current approach the significance weight for each representative is computed as the weighted sum of all the object weights in the partition. However, significance weight can be computed using other numeric aggregation functions.

Examples Continued: Generating Domain-Level Aggregate Profiles

To illustrate the semantic aggregation process, let us return to our movie site example. The aggregation process requires that a "combination function " be defined for each attribute of an object in the domain ontology. Figure 6 and 8 show an example of such process. Each movie object has attribute "Name", "Actor", "Genre" and "Year". For the attribute *Name,* we are interested in all the movie names appearing in the instances. Thus we can define ψ_{Name} to be the union operation performed on all the singleton *Name* attributes of all movie objects. On the other hand, the attribute *Actor* contains a weighted set of objects belonging to class **Actor.** In fact, it represents the relation "Starring" between the actor objects and the movie object. In such cases we can use a vector-based weighted mean operation as the combination function. For example, we will determine the aggregate weight of an actor object *o* by: $w_o^{'} = \left(\sum_i w_i \cdot w_o\right) / \sum_i w_i$.

Applying ψ_{Actor} in our example will result in the aggregate actor object $\{\langle S,0.58\rangle,\langle T,0.27\rangle,\langle U,0.09\rangle\}$. As for the attribute *Year,* the combination function may create a range of all the *Year* values appearing in the objects. Another possible solution is to discretize the full *Year* range into decades and find the most common decades that are in the domains of the attribute. In our example, using the range option, this may result in an aggregate instance [1999, 2002] for the *Year* attribute.

The attribute *Genre* of **Movie** contains a partial order representing a concept hierarchy among different *Genre* values. The combination function, in this case, can perform tree (or graph) matching to extract the common parts of the conceptual hierarchies among all instances. Extracting the common nodes from this hierarchy may also depend on the weights associated with the original objects leading to different weights on the graph edges. For example, given that the higher weight Movies 1 and 2 have "Romance" in common, this node may be selected for the aggregate instance, even though it is not present in Movie 3. However, the weight of "Romance" may be less than that of "Comedy" which is present in all three movies.

Figure 6 shows the item-level usage profile and its representation as a weighted set of objects, as well as the resulting domain-level aggregate profile. Note that the original item-level profile gives us little information about the reasons why these objects were commonly accessed together. However, after we characterize this profile at the domain-level, we find some interesting patterns: they all belong to *Genre* "Comedy" (and to a lesser degree "Romance), and the actor *S* has a high score compared with other actors.

Online Recommendation Phase

In contrast to transaction-based usage profiles, semantic usage profiles capture the underlying common properties and relations among those objects. This fine-grained domain knowledge, captured in aggregate form enables more powerful approaches to personalization. As before, we consider the browsing history of the current user, i.e., active session, to be a weighted set of Web pages that the user has visited. The same transformation described in the last subsection can be used to create a semantic representation of the user's active session. We call this representation the *current user profile.*

Figure 7 presents the basic procedure for generating recommendations based on semantic profiles. The recommendation engine matches the current user profile against the discovered domain-level aggregate profiles. The usage profiles with matching score greater than some pre-specified threshold are considered to represent this user's potential interests. A successful match implies that the current user shares common interests with the group of users represented by the profile. The matching process results in an *extended user profile* which is obtained by applying the aggregation process described above to the domain-level profiles and the original user profile.

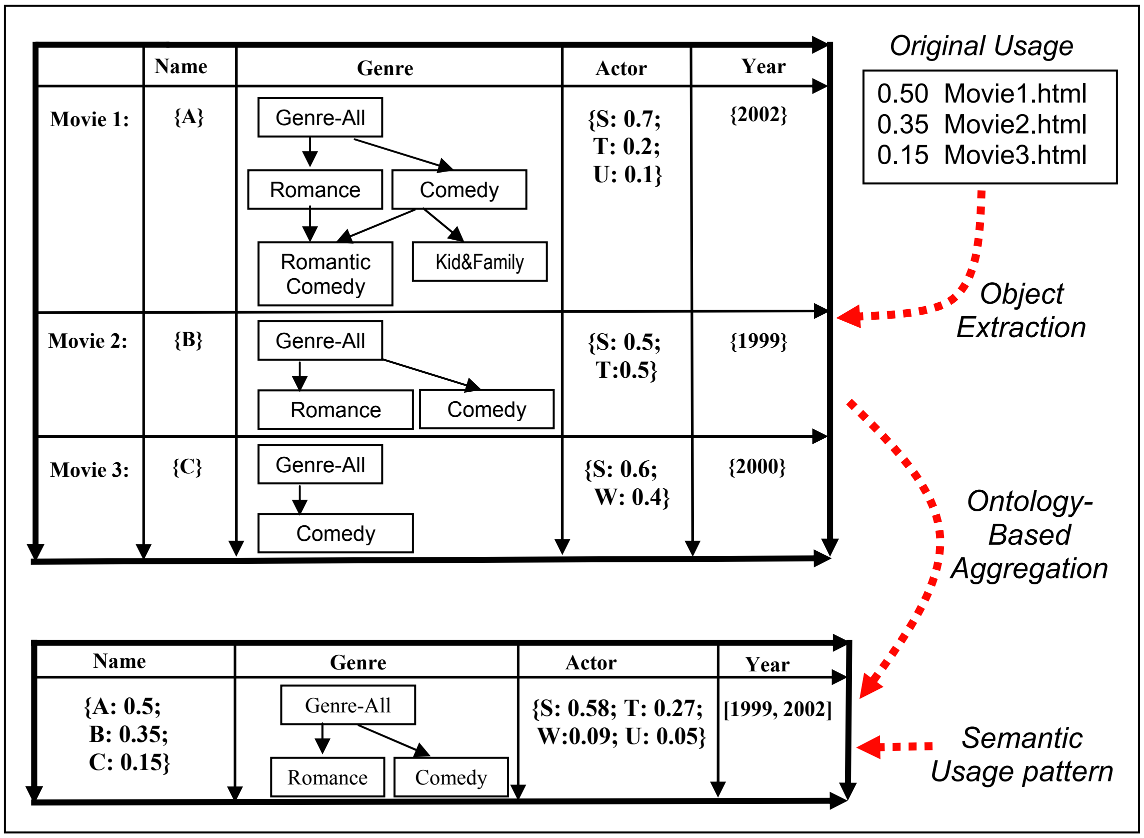


Figure 6: Creating an Aggregate Representation of a Set of Movie Objects

The recommendation engine then instantiates the user's extended profile to real Web objects and will recommend them to the user. We can also exploit structural relationships among classes during the recommendation process. For example, if a concept hierarchy exists among objects, and the recommendation engine can not find a good match for a user profile at a certain concept level, then it can generalize to a more abstract level (e.g., from "romantic comedy" to "romance").

This approach has several advantages over traditional usage-based personalization. First, it retains the user-to-user relationships that can be captured by the discovered usage profiles. Secondly, in contrast to standard collaborative filtering, it provides more flexibility in matching aggregate usage profiles with the current user's activity because the matching process involves comparison of features and relationships, not exact item identities. Thirdly, the items do not have to appear in any usage profiles in order to be recommended, since fine-grained domain relationships are considered during the instantiation process. The previous example shows that this approach can also be used to solve the "new item" problem. Furthermore, it can alleviate the notorious "sparsity" problem in collaborative filtering systems by allowing for "fuzzy" comparisons between two user profiles (or ratings). The basis for matching profiles does not have to be similar ratings on the same items. The comparison can be based on showing interest in different objects with similar properties (for example, purchasing items that have same brand). Therefore, even if the raw transaction or rating data is sparse, the semantic attributes of items or users can be used to indirectly infer potential interest in other items.

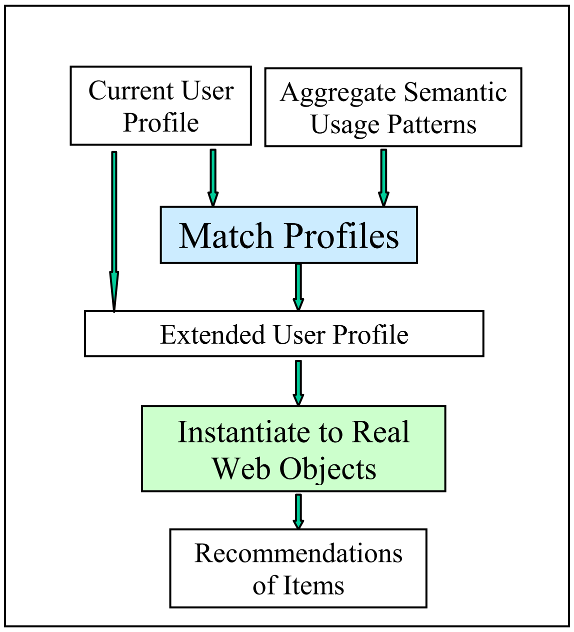


Figure 7: Online Recommendation Enhanced by Domain Ontologies

CONCLUSIONS

We have explored various approaches, requirements, and issues for integrating semantic knowledge into the personalization process based on Web usage mining. We have considered approaches based on the extraction of semantic features from the textual content contained in a site and their integration with Web usage mining tasks and personalization both in the pre-mining and the post-mining phases of the process. We have also presented a framework for Web personalization based on full integration of domain ontologies and usage patterns. The examples provided throughout this chapter reveal how such a framework can provide insightful patterns and smarter personalization services.

We leave some interesting research problems for open discussion and future work. Most important among these are techniques for computing similarity between domain objects and aggregate domain-level patterns, as well as learning techniques to automatically determine appropriate combination functions used in the aggregation process.

More generally, the challenges lie in the successful integration of ontological knowledge at every stage of the knowledge discovery process. In the preprocessing phase, the challenges are in automatic methods for the extraction and learning of the ontologies and in the mapping of users' activities at the clickstream level to more abstracts concepts and classes. For the data mining phase, phase the primary goal is to develop new approaches that take into account complex semantic relationships such as those present in relational databases with multiple relations. Indeed, in recent years, there has been more focus on techniques such as those based relational data mining. Finally in the personalization stage, the challenge is in developing techniques that can successfully and efficiently measure semantic similarities among complex objects (possibly from different ontologies).

In this chapter we have only provided an overview of the relevant issues and suggested a road map for further research and development in this area. We believe that the successful integration of semantic knowledge with Web usage mining is likely to lead to the next generation of personalization tools which are more intelligent and more useful for Web users.

REFERENCES

Anderson, C., Domingos, P., & Weld, D. (2002). Relational markov models and their application to adaptive web mnavigation. *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002)*. Edmonton, Alberta, Canada.

Berendt, B., Hotho, A., & Stumme, G. (2002). Towards semantic web mining. *Proceedings of the First International Semantic Web Conference (ISWC02)*. Sardinia, Italy.

Berendt, B., & Spiliopoulou, M. (2000). Analysing navigation behaviour in web sites integrating multiple information systems. *VLDB Journal, Special Issue on Databases and the Web*, 9(1), 56-75.

Bollacker, K., Lawrence, S., & Giles, C. L. (1998). Citeseer: An autonomous web agent for automatic retrieval and identification of interesting publications. *Proceedings of the Second International Conference on Autonomous Agents*. Minneapolis, Minnesota.

Chakrabarti, S., Dom, B., Gibson, D., Kleinberg, J., Raghavan, P., & Rajagopalan, S. (1998). Automatic resource list compilation by analyzing hyperlink structure and associated text. *Proceedings of the 7th International World Wide Web Conference*. Brisbane, Australia.

Chakrabarti, S., van den Berg, M., & Dom, B. (1999). Focused crawling: A new approach to topic-specific web resource discovery. *Proceedings of the 3th World Wide Web Conference*. Toronto.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. *Proceedings of the ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, California.

Clerkin, P., Cunningham, P., & Hayes, C. (2001). Ontology discovery for the semantic web using hierarchical clustering. *Semantic Web Mining Workshop at ECML/PKDD-2001*. Freiburg, Germany.

Cooley, R., Mobasher, B., and Srivastava, J. (1997). Web mining: information and pattern discovery on the world wide web. *Proceedings of the International Conference on Tools with Artificial Intelligence*, 558-567, Newport Beach. IEEE Press.

Cooley, R., Mobasher, B., & Srivastava, J. (1999). Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems*, 1(1).

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S. (2000). Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1-2), 69-113.

Dai, H., & Mobasher, B. (2002). Using ontologies to discover domain-level web usage profiles. *Proceedings of the 2nd Semantic Web Mining Workshop at ECML/PKDD 2002*. Helsinki, Finland.

Ganesan, P., Garcia-Molina, H., & Widom, J. (2003). Exploiting hierarchical domain structure to compute similarity. *ACM Transactions on Information Systems*, *21*(1), 63-94.

Getoor, L., Friedman, N., Koller, D., & Taskar, B. (2001). Learning probabilistic models of relational structure. *Proceedings of the 18th International Conference on Machine Learning*.

Ghani, R., & Fano, A. (2002). Building recommender systems using a knowledge base of product semantics. *Proceedings of the Workshop on Recommendation and Personalization in E-Commerce*, at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems. Malaga, Spain.

Giugno, R., & Lukasiewicz, T. (2002). P-SHOQ(D): A probabilistic extension of SHOQ(D) for probabilistic ontologies in the semantic web. *Proceedings of the 8th European Conference on Logics in Artificial Intelligence*. Cosenza, Italy.

Han, J., & Fu, Y. (1995). Discovery of multiple-level association rules from large databases. *Proceedings of the 1995 Int'l Conf. on Very Large Data Bases (VLDB95).* Zurich, Switzerland.

Herlocker, J., Konstan, J., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. *Proceedings of the 22nd ACM Conference on Research and Development in Information Retrieval (SIGIR'99)*. Berkeley, CA.

Horrocks, I. (2002). DAML+OIL: A description logic for the semantic web. *IEEE Data Engineering Bulletin*, 25(1), 4-9.

Horrocks, I., & Sattler, U. (2001). Ontology reasoning in the SHOQ(D) description logic. *Proceedings of the 17th International Joint Conference on Artificial Intelligence*. Seattle, WA.

Hotho, A., Maedche, A., & Staab, S. (2001). Ontology-based text clustering. *Proceedings of the IJCAI-2001 Workshop Text Learning: Beyond Supervision*. Seattle, WA.

Jin, X., Mobasher, B. (2003). Using semantic similarity to enhance item-based collaborative filtering. *Proceedings of The 2nd IASTED International Conference on Information and Knowledge Sharing*. Scottsdale, Arizona.

Joachims, T., Freitag, D., and Mitchell, T. (1997). Webwatcher: A tour guide for the world wide web. *Proceedings of the 15th International Conference on Artificial Intelligence*, Nagoya, Japan.

Lieberman, H. (1995). Letizia: An agent that assists web browsing. *Proceedings of the 1995*

International Joint Conference on Artificial Intelligence, Montreal, Canada.

Loh, S., Wives, L. K., & de Oliveira, J. P. (2000). Concept-based knowledge discovery in texts extracted from the web. *SIGKDD Explorations*, 2(1), 29-39.

Maedche, A., Ehrig, M., Handschuh, S., Volz, R., & Stojanovic, L. (2002). Ontology-focused crawling of documents and relational metadata. *Proceedings of the 11th International World Wide Web Conference (WWW02)*. Honolulu, Hawaii.

Maedche, A., & Staab, S. (2000). Discovering conceptual relations from text. *Proceedings of the European Conference on Artificial Intelligence (ECAI00)*. Berlin.

McCallum, A., Rosenfeld, R., Mitchell, T., & Ng, A. (1998). Improving text classification by shrinkage in a hierarchy of classes. *Proceedings of the 15th International Conference on Machine Learning*. Madison, Wisconsin.

Melville, P., Mooney, R. J., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI02)*. Edmonton, Alberta, Canada.

Mobasher, B., Cooley, R., & Srivastava, J. (2000a). Automatic personalization based on web usage mining. *Communications of the ACM*, 43(8), 142-151.

Mobasher, B., Dai, H., Luo, T., & Nakagawa, M. (2001). Effective personalization based on association rule discovery from web usage data. *Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01).* Atlanta, Georgia.

Mobasher, B., Dai, H., Luo, T., Sun, Y., & Zhu, J. (2000b). Integrating web usage and content mining for more effective personalization. *E-Commerce and Web Technologies: Proceedings of the EC-WEB 2000 Conference*, Lecture Notes in Computer Science (LNCS) 1875, 165-176. Springer.

Mobasher, B., Dai, H., Luo, T., & Nakagawa, M. (2002). Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining and Knowledge Discovery*, 6, 61-82.

Palopoli, L., Sacca, D., Terracina, G., & Ursino, D. (2003). Uniform techniques for deriving Similarities of objects and subschemes in heterogeneous databases. *IEEE Transactions on Knowledge and Data Engineering*, *15*(1), 271-294.

Palpanas, T. and Mendelzon, A. (1999). Web prefetching using partial match prediction. *Proceedings of the 4th International Web Caching Workshop (WCW99)*. San Diego, CA.

Parent, S., Mobasher, B., & Lytinen, S. (2001). An adaptive agent for web exploration based on concept hierarchies. *Proceedings of the International Conference on Human Computer Interaction*. New Orleans, LA.

Pazzani, M. (1999). A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13 (5-6), 393-408.

Perkowitz, M. and Etzioni, O. (1998). Adaptive Web sites: automatically synthesizing Web pages. *Proceedings of the 15th National Conference on Artificial Intelligence*. Madison, WI.

Pitkow, J. and Pirolli, P. (1999). Mining longest repeating subsequences to predict WWW surfing. *Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems*. Boulder, Colorado.

Rodriguez, M. A. & Egenhofer, M. J. (2003). Determining semantic similarity among entity classes from different ontologies. *IEEE Transactions on Knowledge and Data Engineering*, *15*(2), 442-456.

Sarwar, B. M., Karypis, G., Konstan, J., & Riedl, J. (2000). Analysis of recommender algorithms for e-commerce. *Proceedings of the 2nd ACM E-Commerce Conference (EC'00).*

Minneapolis, MN.

Srivastava, J., Cooley, R., Deshpande, M., & Tan, P. (2000). Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, 1(2), 12-23.

Spiliopoulou, M. (2000). Web usage mining for Web site evaluation. *Communications of ACM*. 43(8), 127-134.

Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., & Lakhal, L. (2000). Fast computation of concept lattices using data mining techniques. *Proceedings of the Knowledge Representation Meets Databases Conference (KRDB00)*. Berlin.